PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Application No.:	10/660,029	Confirmation No. 5809
Applicant:	Wen-Hua Lin	
Filed:	September 11, 2003	
Art Unit:	2191	
Examiner:	Anna C. Deng	
Atty. Dkt. No.:	7196-128/10311067	

==========

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

AMENDMENT

In response to the November 1, 2006 Office Action, please amend the above-identified application as follows:

Amendments to the Specification begin on page 2 of this paper.

Amendments to the Claims are reflected in the listing of claims which begins on page 4 of this paper.

Remarks/Arguments begin on page 6 of this paper.

Amendments to the Specification

1. Please replace the paragraph beginning at page 1, line 4, that starts with "This invention relates to information", with the following amended paragraph:

This invention relates to information technology, and more particularly, to an embedded system program code reduction method and system, which is used for scaling down the total amount of program code that is to be burned into an embedded system for the purpose of reducing the embedded system's memory requirement and also allowing the embedded system to be increased in performance. The program code that is to be scaled down includes a virtual machine, such as ~~Java~~ JAVA Virtual Machine (JVM) or ~~Microsoft~~ MICROSOFT Virtual Machine (MVM), and a set of application programs running on the virtual machine.

2. Please replace the paragraph beginning at page 1, line 12, that starts with "Portable information platforms, such as", with the following amended paragraph:

Portable information platforms, such as mobile phones, PDAs, pagers, etc., are typically based on an embedded controller that integrates a microprocessor and a set of system and application programs in the same device. Presently, a virtual machine, such as ~~Java~~ JAVA Virtual Machine (JVM) or ~~Microsoft~~ MICROSOFT Virtual Machine (MVM) is integrated to the embedded system as a cross-platform foundation for the running of application programs on the information platform. The use of virtual machine allows application programs to have the so-called "write once, run everywhere" capability that allows one version of an application program to run on different types of information platforms installed with different kinds of operating systems. This feature allows software programmers to write just one version of an application program of a specific function, and the application program can then run on different types of information platforms installed with different kinds of operating systems, without having to write many different versions of applications programs for the same functionality.

3. Please replace the paragraph beginning at page 4, line 12, that starts with "The embedded system program code reduction", with the following amended paragraph:

The embedded system program code reduction method and system according to the invention is designed for use to scale down the total amount of program code that is to be burned into an embedded system for the purpose of reducing the embedded system's memory requirement and also allowing the embedded system to be increased in performance. The program code that is to be scaled down includes a virtual machine, such as ~~Java~~ JAVA Virtual Machine (JVM) or ~~Microsoft~~ MICROSOFT Virtual Machine (MVM), and a set of application programs running on the virtual machine.

4. Please replace the paragraph beginning at page 6, line 10, that starts with "The virtual machine 20", with the following amended paragraph:

The virtual machine 20 is, for example, a ~~Java~~ JAVA Virtual Machine (JVM) or a ~~Microsoft~~ MICROSOFT Virtual Machine (MVM), which includes an object library 21, a complier 22, and a runtime environment 23. Since JVM and MVM are well-known virtual machines in the information industry, detailed description thereof will not be given here in this specification

Amendments to the Claims

This listing of claims will replace all prior versions, and listings, of the claims in the application:

Listing of Claims

1. (currently amended) An embedded system program code reduction method for scaling down a virtual machine and a set of application programs running on the virtual machine that are to be burned into an embedded system, wherein the virtual machine includes an object library, a complier, and a runtime environment;

the embedded system program code reduction method comprising:

a compilation procedure for compiling the source code of each application program into bytecode;

an object picking procedure for picking ~~those~~ from the object library essential objects that are required for use by the application programs during runtime and collectively ~~pack~~ packing all picked objects into an essential-objects package

a compression procedure for compressing the essential-objects package into a compressed file of essential objects; and

a code integration procedure for integrating each bytecode-based application program, the compressed file of essential objects ~~from the compression module~~, and the runtime environment from the virtual machine into a set of embedded system program code which is to be burned into the embedded system.

2-3. (canceled)

4. (currently amended) An embedded system program code reduction system for use to scale down a virtual machine and a set of application programs running on the virtual machine that are to be burned into an embedded system, wherein the virtual machine includes an object library, a complier, and a runtime environment;

the embedded system program code reduction system comprising:

a compilation module, which is used to compile the source code of each application program into bytecode;

an object picking module, which is used to pick ~~those~~ from the object library essential objects that are required for use by the application programs during runtime and collectively pack all picked objects into an essential-objects package;

a compression module, which is used to compress the essential-objects package into a compressed file of essential objects; and

a code integration module, which is used to integrate each bytecode-based application program, the compressed file of essential objects from the compression module, and the runtime environment from the virtual machine into a set of embedded system program code which is to be burned into the embedded system.

5-6. (canceled)

REMARKS/ARGUMENTS

The specification is amended to recognize the trademark JAVA and MICROSOFT.

Claims 1-6 are pending.

Claims 1 and 4 are amended to replace references to "those" with the phrase "from the object library." Claim 1 is also amended to remove the reference to "compression module." These amendments overcome the 35 USC § 112 objections to claims 1 and 4.

It is noted that an antecedent basis for the phrase "the compression module" is present in claim 4 (see line 11, reciting "a compression module…")

Claim 1 is also amended to replace "pack" with "packing."

Claims 2, 3, 5 and 6 are canceled without prejudice to pursue these claims in other applications. Cancellation renders moot the 35 USC § 112 objections to these claims.

The rejection of claims 1 and 4 as anticipated by published patent application No. 2003/0009743 of Fresko et al. ("Fresko") is respectfully traversed. "To anticipate a claim, the reference must teach every element of the claim." MPEP § 2131. In the present case, Fresko fails to achieve this standard.

The present invention is directed to an embedded system program code reduction method and system. As recited in amended claims 1 and 4, the method, and the system as well, first compiles the source code of each application program into bytecode, then picks from the object library essential objects that are required for use by the application programs during runtime and compresses the picked essential objects, and lastly integrates each bytecode-based application program with the compressed essential objects into a set of embedded system program code which is to be burned into the embedded system.

Fresko discloses a method and apparatus for preprocessing and packaging JAVA class files. The method removes duplicate information elements from a set of class files to reduce the size of individual class files. In practice, Fresko teaches, first, examining whether the constant pool table 305 of the class files matches another constant pool table of other class files, then removing the constant pool table 305 of the class files if the constant pool table 305 of the class files matches the another constant table of the other class files, compressing the class files without the removed constant pool table 305, and generating a shared constant pool table, and lastly integrating the shared constant pool table with the class files without the removed constant pool table 305.

The Office Action alleges that Fresko discloses in FIG. 4 and paragraphs [0062] and [0063] of the specification the technical feature of "picking essential objects that are required for use by the application programs during runtime," as recited in amended claims 1 and 4. However, Fresko discloses in FIG. 4 and paragraphs [0062] and [0063] the technical feature of "examining whether the constant pool table 305 of the class files matches another constant pool

table of other class files" and "removing the constant pool table 305" only, but fails to teach or suggest the technieal feature of "picking essential objects that are required for use by the application programs during runtime."

Further, the Office Action alleges that Fresko discloses in FIG. 4 and paragraph [0065] the technical feature of "compressing the picked essential objects," as recited in amended claims 1 and 4. However, Fresko discloses in FIG. 4 and paragraph [0065] the technical feature of "compressing the class files with the constant pool table 305 removed" only, but fails to teach or suggest the technical feature of "compressing the picked essential objects."

The Office Action further alleges that Fresko discloses in FIGS. 4 and 5 and paragraphs [0065] and [0068] of the specification the technical feature of "integrating each bytecode-based application program with the compressed essential objects into a set of embedded system program code," as recited in amended claims 1 and 4. However, Fresko discloses in FIGS. 4 and 5 and paragraphs [0065] and [0068] the technical feature of "integrating the shared constant pool table with the class files with the eonstant pool table 305 removed" only, but fails to teach or suggest the technical feature of "integrating each bytecode-based application program with the compressed essential objects into a set of embedded system program code."

Because Fresko fails to teach or suggest all elements of the claims, claims 1 and 4 are not anticipated by this reference.

In view of the foregoing amendments and remarks, Applicant submits that the present application is in condition for allowance. A Notice of Allowance is therefore respectfully requested.

No fee is believed due. However, the Commissioner is hereby authorized during prosecution of this application and any related appeal, to charge any fees that may be required (except for patent issue fees required under 37 CFR §1.18) or to credit any overpayment of fees to Deposit Account No. 50-0337. If an extension of time is required in connection with this paper, please consider this a Petition therefor and charge any fees required to Deposit Account No. 50-0337.

Dated: January 27, 2007

Respectfully submitted,

Miles Yamanaka

Miles Yamanaka
Reg. No. 45,665

FULBRIGHT & JAWORSKI L.L.P
555 South Flower Street , 41st Floor
Los Angeles, CA 90071
(213) 892-9200 – Telephone
(213) 892-9494 – Facsimile